Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application.

Listing of Claims:

1 (currently amended). A method of executing instructions in a microprocessor comprising:

A4

fetching a conditional branch instruction from an instruction cache;

detecting branch prediction information embedded in the branch instruction, wherein the embedded branch prediction information is indicative of the likelihood of successfully predicting the result of the conditional branch instruction; and

responsive to the branch prediction information indicating that the conditional branch instruction is not likely to be successfully predicted, fetching instructions from both a branch-taken path and from a branch-not-taken path of the branch instruction.

2 (original). The method of claim 1, further comprising:

speculatively executing the instructions from the branch-taken path and the branch-not-taken path of the branch instruction;

executing the conditional branch instruction; and

based upon the outcome of the conditional branch instruction, discarding results from the speculatively executed instructions from the branch-taken path if the branch is not taken and discarding results from the branch-not-taken path if the branch is taken.

3 (currently amended). The method of claim 1, wherein detecting the branch prediction information comprises detecting compiler generated branch prediction information ~~[indicative of the context in which the conditional branch instruction is used]~~.

4 (currently amended). The method of claim 3, wherein the branch prediction information causes instruction fetching from both the taken and not taken branches if a compiler determines that the probability of successfully predicting the conditional branch instruction is ~~[determined by the compiler to be unpredictable]~~ less than approximately 75%.

5 (canceled).

6 (original). The method of claim 1, wherein fetching instructions from the branch-not-taken path comprises fetching instructions down the branch-not-taken path until a subsequent branch instruction is encountered.

A4

7 (currently amended). A microprocessor comprising:

an instruction cache suitable for storing a set of processor executable instructions and configured to receive an instruction address and to retrieve an instruction corresponding to the instruction address; and

a fetch unit connected to the instruction cache and configured to generate an instruction address;

wherein the fetch unit is configured to detect branch instruction information embedded in a branch instruction retrieved from the instruction ~~[cache]~~ cache, wherein the branch prediction information indicates whether the branch instruction is likely to be successfully predicted, and further configured to fetch instructions from both a branch-taken path and a branch-not-taken path of the branch instruction ~~[depending upon the state of the branch instruction information]~~ if the branch prediction information indicates that the branch instruction is not likely to be predicted successfully.

8 (currently amended). The microprocessor of claim 7, [~~wherein the~~] further comprising a branch prediction unit [~~includes~~] comprising prediction logic ~~[enabled]~~ invoked by the branch instruction information only if the embedded branch prediction information indicates that the branch instruction is likely to be predicted successfully and configured to predict the result of a branch instruction based on information in a branch history table.

9 (currently amended). The microprocessor of claim 8, wherein the branch [~~predication~~] prediction unit includes a prediction bypass unit ~~[enabled by]~~ invoked only if the branch prediction information indicates that the branch instruction is not likely to be predicted successfully and configured to issue instruction addresses from a branch-taken path and a branch-not-taken path of the branch instruction.

10 (original). The microprocessor of claim 7 wherein the processor is configured to speculatively execute the instructions from the branch-taken path and from the branch-not-taken path of the branch instruction, execute the conditional branch instruction, and, based upon the outcome of the conditional branch instruction, discard results from the speculatively executed instructions from the branch-taken path if the branch is not taken and discarding results from the branch-not-taken path if the branch is taken.

11 (currently amended). The microprocessor of claim 7 ~~[configured to receive]~~ wherein the branch prediction information comprises compiler generated information ~~[indicative of the context in which the conditional branch instruction is used]~~.

12 (currently amended). The microprocessor of claim 7, wherein the microprocessor is configured to fetch a predetermined number of instructions from the branch-taken path and further configured to fetch ~~[a]~~ the same predetermined number of instructions from the branch-not-taken path depending upon the state of the branch instruction information.

A4

13 (original). The microprocessor of claim 7, further configured ~~[for fetching]~~ to fetch instructions down the branch-not-taken path until a subsequent branch instruction is encountered depending upon the state of the branch instruction information.

14 - 19. (canceled).

20 (new). The method of claim 4, further comprising, responsive to the branch prediction information indicating that the conditional branch instruction is likely to be successfully predicted, using a branch history table to predict the branch and, based upon the prediction, fetching instructions from only the predicted path.

21 (new). A compiler, comprising:

compiler means for detecting a conditional branch instruction;

compiler means for determining the likelihood of successfully predicting the branch instruction; and

compiler means for embedding branch instruction information within the branch instruction, wherein the branch instruction information is indicative of the likelihood of successfully predicting the branch instruction.

22 (new). The compiler of claim 21, wherein the means for embedding branch instruction information within the branch instruction comprises means for embedding a first branch instruction information code in the branch instruction if the branch instruction is not likely to be predicted successfully and a second branch instruction code otherwise.